Surrogate-assisted evolutionary biobjective optimization for objectives with non-uniform latencies

Tinkle Chugh[1,2], Richard Allmendinger[3], Vesa Ojalehto[2], and Kaisa Miettinen[2]

[1]Department of Computer Science, University of Exeter, United Kingdom
[3]The University of Manchester, Manchester Business School,, United Kingdom
[2]University of Jyvaskyla, Faculty of Information Technology, Finland Faculty of Information Technology, P.O. Box 35 (Agora), FI-40014, University of Jyvaskyla, Finland

Abstract

We consider multiobjective optimization problems where objective functions have different (or heterogeneous) evaluation times or latencies. This is of great relevance for (computationally) expensive multiobjective optimization as there is no reason to assume that all objective functions should take an equal amount of time to be evaluated (particularly when objectives are evaluated separately). To cope with such problems, we propose a variation of the Kriging-assisted reference vector guided evolutionary algorithm (K-RVEA) called heterogeneous K-RVEA (short HK-RVEA). This algorithm is a merger of two main concepts designed to account for different latencies: A single-objective evolutionary algorithm for selecting training data to train surrogates and K-RVEA's approach for updating the surrogates. HK-RVEA is validated on a set of biobjective benchmark problems varying in terms of latencies and correlations between the objectives. The results are also compared to those obtained by previously proposed strategies for such problems, which were embedded in a non-surrogate-assisted evolutionary algorithm. Our experimental study shows that, under certain conditions, such as short latencies between the two objectives, HK-RVEA can outperform the existing strategies as well as an optimizer operating in an environment without latencies.

Keywords: Metamodelling, multiobjective optimization, expensive optimization, Pareto optimality, heterogeneous objectives, machine learning, Bayesian optimization

1 Introduction

In many real-world multiobjective optimization problems (MOPs), the computation time for evaluating objective functions can be substantial. For instance, evaluations involving computational fluid dynamics simulations [10] or real physical, biological or chemical experiments [20] can be very time-consuming (i.e. computationally or otherwise expensive). Therefore, it is desirable to obtain solutions within few evaluations. In the last few years, surrogate-assisted approaches embedding evolutionary algorithms have been proposed for such problems, referred to as surrogate-assisted evolutionary algorithms (SAEAs) in this article. These algorithms train surrogates to replace expensive functions, efficiently select the training data and optimize the surrogate with an optimizer, e.g., an evolutionary algorithm. Examples of SAEAs for MOPs include ParEGO [15] and K-RVEA [7]. For more details, e.g., characteristics, advantages and limitations of SAEAs and general overviews, see, e.g., [1, 8].

As shown, e.g., in a recent survey [8], all SAEAs proposed in the literature assume the same evaluation time for all objective functions, which is not always realistic for MOPs. This is especially the case when objectives are evaluated separately as opposed to, e.g., doing a single expensive experiment (e.g., a simulation) and then extracting objectives from the simulation results afterwards. For instance, in finding the optimal process conditions of a chemical reactor [19] and designing a wastewater treatment operation [13], one of the decision variables is also considered as an objective function but other objectives necessitate calling a time-consuming simulator. Similarly, in a drug discovery problem [20], one objective requires a real time-consuming physical experiment to

measure the potency of a drug cocktail, while another objective represents simply the number of drugs in a drug cocktail. We use the term *latency* to denote the ratio of the computation time of one evaluation of the most expensive objective function to the computation time of one evaluation of another (cheaper) objective function.

Applying conventional SAEAs for solving MOPs with different latencies among objective functions may not be a viable option, unless such latencies are ignored. In this article, we modify the Kriging-assisted reference vector guided evolutionary algorithm (K-RVEA) [7] to be applicable for expensive problems having different latencies. K-RVEA is a recently proposed surrogate-based evolutionary algorithm for optimization problemw with more than 3 objectives. The efficiency of the algorithm has been demonstrated and compared with other algorithms in [7]. In particular, we employ surrogates and address the following challenges of considering different computation times of objective functions:

1. how to make the most of the function evaluations that are affordable,
2. how to select training data, and
3. how to update the surrogate models?

We propose an algorithm called heterogeneous K-RVEA (HK-RVEA) which uses Kriging models with a Gaussian kernel (implemented in the DACE library [18]) as surrogates and makes the best use of function evaluations available, especially for a less expensive (i.e., cheap) objective function. In addition, the algorithm uses single-objective evolutionary algorithm (EA) when selecting the training data and efficiently updates the surrogates with a strategy from K-RVEA. The strategy selects samples based on the improvement in the objective function values by using a criterion called angle penalized distance and uncertainty information from Kriging models. We test the new algorithm on several biobjective benchmark optimization problems with different latencies. Moreover, correlations among different objective functions are also taken into account.

The rest of this paper is structured as follows. In the next section, we survey recent studies in the field and introduce the HK-RVEA algorithm in Section 3. Results of numerical experiments on several benchmark problems with different latencies and correlations between objective functions are presented in Section 4. Finally, we conclude and discuss future research directions in Section 5.

2 Related work

In general, optimization algorithms assume that, for a given solution, the objective (and constraint) functions of a problem can be evaluated simultaneously, i.e., without any delays. However, this does not necessarily need to be the case, especially, when dealing with real-world problems as discussed in the introduction. In the following, we briefly review some related literature and link it with the focus of this paper.

The most relevant literature to our research was reported in [2, 3], where a formal introduction was provided to optimization problems subject to latencies in the objective functions and strategies were proposed for such problems. These strategies will be explained in more detail in the next section. In [3], a first attempt was also made to look at approximation-based strategies using a simple fitness inheritance-based method. It was shown that the presence of latencies on the objective functions affects the performance of an optimizer in general, and that both the length of the latency and the correlation between the objectives to be optimized are factors to be considered in the algorithm design. We extend the work reported in [2, 3] by exploring the application of a surrogate-assisted approach to cope with latencies in HK-RVEA. Moreover, while the previous work considered binary optimization problems, we extend the study to continuous problems.

Surrogate-assisted optimization has become an accepted approach for dealing with expensive optimization problems, and merging it with EAs (known as SAEAs) has become a powerful methodology that has led to many success stories [1, 8]. While initial research in the field of surrogate-assisted optimization has been focused on unconstrained single-objective optimization problems, there have been attempts of extending the methodology to take into account problem features such as constraints [4, 9] and many objectives [7]. This is the first research that extends SAEAs to deal with multiobjective optimization problems that have different latencies in the objectives. To make SAEAs work on such problems, a key aspect to address is how to consider accurate objective

function values (obtained by evaluating the real objective functions) and approximate objective function values (obtained by the surrogate) in the selection of samples and utilizing available computation time.

In the following, we link the key steps of the proposed approach with related techniques without going into too many details. This facilitates a better understanding of HK-RVEA to be introduced in the subsequent section.

We consider biobjective optimization problems and a key step of HK-RVEA is to use up the whole budget available for both objective functions by interleaving evaluations on the expensive and the cheap to evaluate objective function. We achieve this by running a single-objective EA on the cheap objective while the expensive objective function is being evaluated. The information gathered about the cheap objective function is then used to make an informative decision about which solution to evaluate next on the expensive objective function. This approach is similar to brood selection [22], which first uses a variation to create more offspring than needed and then applies a filtering step to eliminate a subset of them (to maintain a constant population size).

The combination of having latencies in the objectives and an interleaving evaluation scheme means that the proposed approach needs to operate on a population that consists of solutions that have been evaluated on both objectives or one (the cheap) objective only. Also, these solutions are typically generated by parents that vary in the number of generations they have evolved. This situation is similar to the idea of maintaining age-layered populations [14], and inspiration can be taken from this work to perform selection and population updates. The absence of objective values is comparable to missing data in classical machine learning problems, which can be circumvented using imputation methods [21].

The fact that the number of solutions evaluated on both objectives or only on one (the cheap one) varies means that training the surrogates is done on two different sets (more sets may be needed if more than two objectives are optimized). This setting is similar to using an ensemble of models for training [17].

Another area of research related to ours is simplifying a multiobjective optimization problem by removing redundant objectives [5]. While the effect of objective function removal may seem similar to the effect of latencies in objectives, there are clear differences. Our goal is not to identify which objectives we can neglect but rather to estimate the effects of neglecting one expensive objective to decide whether it is worthwhile to use the objective.

An application area that shares similarities with latencies in objectives is asynchronous evaluations in distributed computer grids or non-dedicated peer-to-peer systems [16]. In such environments, an optimization algorithm may not be able to afford waiting for all solutions in a population to be evaluated before continuing with the evolution. Instead, offspring may need to be generated prematurely, e.g., once a desired number of solutions has been evaluated. To some sense, this happens in an interleaved scheme for coping with latencies in the objectives. However, more importantly, in our case the asynchrony is also present across the objectives of a single solution and, moreover, depending on the latency, more or less solutions have missing information about their (expensive) objective function values.

3 Heterogeneous surrogate-assisted reference vector guided evolutionary algorithm (HK-RVEA)

The HK-RVEA algorithm uses Kriging models as surrogates and has three major features:

1. selection of training data set,
2. updating surrogate models and
3. using single-objective optimization.

The first two features are common challenges of developing a SAEA. However, they are more severe when the computation cost of objective functions is different i.e., objective functions have different latencies. HK-RVEA efficiently selects the samples for training and updating the surrogates considering different latencies of the objective functions. In addition, it uses single-objective optimization in selecting samples for training the surrogates. The steps are given in Algorithm 1 and described in more detail below.

Before going to the details of the steps, however, we introduce some notation and assumptions made. Assuming a biobjective optimization problem throughout this paper, we use the following notation:

1. objective function vector $f = (f_1, f_2)$
2. f_i^{ex} = the more time-consuming objective function (w.l.o.g. this will be fixed to f_2 in the experimental study)
3. f_j^{nex} = the cheap (or fast) to evaluate objective function, where $j \neq i$ (this will be f_1 in the experimental study)
4. latency = ratio of the computation time of one evaluation of f_i^{ex} to the computation time of one evaluation of f_j^{nex}, where $j \neq i$.

Algorithm 1: HK-RVEA

Input: $MaxFE^{ex}$ or $MaxFE^{nex}$: maximum number of function evaluations for f_i^{ex} or f_j^{nex} and latency
Output: nondominated solutions of the archive A
1. Create an initial population P generated with some design of experiment technique, set *iteration count = 1* and initialize $FE^{ex} = 0$, $FE^{nex} = 0$, $A = \phi$ and $A^{nex} = \phi$
while $FE^{ex} < MaxFE^{ex}$ *OR* $FE^{nex} < MaxFE^{nex}$ **do**
  while P *is evaluated with* f_i^{ex} **do**
    2. Evaluate f_j^{nex} using P and add the solutions to A^{nex}
    if *iteration count = 1* **then**
      3. Run a single-objective EA (without any surrogate) to optimize f_j^{nex} using $|P|\times$ (latency- 1) function evaluations and add the solutions to A^{nex}
    else
      3. Do crossover and mutation in P to produce a population of size $|P|\times$ (latency- 1) and evaluate them with f_j^{nex} add to A^{nex}
  4. Update $FE^{ex} = FE^{ex} + |\mathrm{P}|$ and $FE^{nex} = FE^{nex} + (|P|\times$ latency) and the archive A with solutions obtained with both objective functions
  5. Build surrogates for both objective functions by selecting a training data set
  6. Run an evolutionary algorithm to find samples for updating the surrogates
  7. Select sample(s) (denoted by P) by using some infill criterion, e.g., from the K-RVEA algorithm and update *iteration count = iteration count + 1*

We make the following assumptions:

1. both objective functions can be evaluated in parallel,
2. for each objective function, samples for training can be evaluated in batches,
3. w.l.o.g. the cheap objective shall take one time unit to be evaluated meaning the latency is simply the number of time units needed to evaluate the expensive objective function (note that a latency of 1 means both objectives take equally long to be evaluated), and
4. computational overhead of running a single-objective EA (e.g., the computation time of using crossover, mutation and selection operators) in step 3 of Algorithm 1 is negligible.

Input: Maximum number of function evaluations for both or one objective function and latency.
Output: Nondominated solutions of an archive A, which stores all evaluated solutions with both objective functions.
Termination criterion: The algorithm terminates once the maximum number of function evaluations of f_i^{ex} or f_j^{nex}, $j \neq i$ is used. For instance, if $MaxFE^{ex}$ is the maximum number of evaluations for f_i^{ex}, then one can devote $MaxFE^{nex} = MaxFE^{ex} \times latency$ evaluations to f_j^{nex}.
Steps:

1: To get started, a predefined number of samples P is generated, e.g., using the Latin hypercube sampling and the number of function evaluations and the iteration count are initialized. Two empty archives A and A^{nex} are defined, which store all the evaluated solutions with both objective functions and the cheap objective function only, respectively.[1]

2: We start evaluating samples with both objective functions. As the evaluation times of the objective functions are different, we assume that we have solutions evaluated with the cheap objective function f_j^{nex} but evaluations of P with the expensive objective function f_i^{ex} are not completed.

3: As the evaluations of f_i^{ex} are still running, we can afford more evaluations (as many as $|P|\times$ (latency- 1) with f_j^{nex}, where $|P|$ represents the size of the sample set P). In this, we use a single-objective EA to find solutions for the cheap objective function. Note that the maximum number of function evaluations (also the termination criterion) for running the single-objective EA is $|P|\times$ (latency- 1). There are two reasons for using the single-objective EA: 1) making the most of the computation time (or number of function evaluations $|P|\times$ (latency- 1)) available and 2) finding promising samples for training the surrogate model of the cheap objective function. The strategy to select training samples is described in step 5.

We do the single-objective optimization only once. This is because the number of function evaluations available after the first iteration (i.e. *iteration count* > 1) is not sufficient for the single-objective optimization (described in step 7). Therefore, we use crossover and mutation to generate samples and evaluate them with the cheap objective function as mentioned in step 3.

4: Thus far, we have solutions evaluated with both objective functions and they are added in the archive A. We also have a different set of solutions evaluated with f_j^{nex} using single-objective EA or crossover and mutation operators, which is stored in the archive A^{nex}. In addition, we update the number of function evaluations with both objective functions.

5: We build the surrogate models for both objective functions. As mentioned in the previous step, we have solutions obtained with different sets of samples (in the decision space) for both objective functions. Therefore, we do not use the same samples for training the surrogates, which differs from conventional surrogate-assisted algorithms. Naturally, the number of solutions in A^{nex} is bigger than the number of solutions in A. In other words, the number of training samples for building the surrogate models is different for both objective functions.

We train the surrogate for f_j^{nex} by using solutions in A^{nex} and for f_i^{ex} by using solutions in A^{ex}. The number of samples in both archives (can be substantial in A^{nex}) increases with the number of evaluations, which can increase the training time for building the surrogates. Therefore, we select a pre-defined maximum number of samples (same for both objective functions) for building the surrogate models. We select a distinct set of samples in the decision space, e.g., by doing k-means clustering, where the cluster size k is equal to the pre-defined maximum number of samples. In this article, the size is selected arbitrarily and needs to be adaptive, which is considered as a future research work. So far, we have selected the training data and built the surrogates for both objective functions.

6: After building the surrogates, we run a multiobjective EA (in our case RVEA [6]) to find samples for updating the surrogates.

7: Samples are selected to update the surrogates. We adopt the strategy from K-RVEA [7], which selects samples based on the needs of the convergence and diversity. For convergence, a criterion called angle penalized distance and for diversity, the uncertainty values obtained from the Kriging models are used in selecting the samples. Moreover, reference vectors are used for the needs of diversity. For more details, see [7]. The selected samples P are then evaluated in step 2. The algorithm is terminated after a maximum number of function evaluations as defined earlier.

[1] Note that the population is not yet evaluated in this step and, therefore, the number of function evaluations is set to zero.

4 Experimental study

This section provides our experimental setup and then demonstrates the proposed HK-RVEA algorithm on several biobjective benchmark problems.

4.1 Experimental setup

In the following we outline the algorithms, test problems and performance metrics used in the experimental study. In particular, the performance of HK-RVEA is compared against the four algorithms introduced in [2]. Section 2 has touched upon these algorithms but here we provide a slightly more in depth description of them.

- Waiting: this method is the default strategy to deal with differing latencies. It simply waits until the evaluation of all objective functions is completed before evaluating the next solution. Hence, this approach is similar to conventional evolutionary multiobjective algorithms.
- Fast-first: this method optimizes the cheap objective for most of the evaluations. It then switches to the expensive objective at the latest possible point to ensure that all objective function values are obtained for at least some solutions (here we ensure that a prefixed number of samples is evaluated on the expensive objective function).
- Speculative (interleaving): this method attempts to utilize the full budget of evaluations available under both the expensive and the cheap objectives. It works similar to the approach used by HK-RVEA in the sense that a single-objective EA is employed to optimize the cheap objective while solutions are evaluated on the expensive objective. These are then used to inform the multiobjective EA about which solutions to evaluate on the expensive objective function next.
- Brood (interleaving): this method works similar to the speculative interleaving method with the difference that, instead of applying a single-objective EA on the cheap objective, solutions are generated by uniform selection and variation with the goal to maintain an unbiased solution set.

The four algorithms above are embedded in an evolutionary multiobjective optimization algorithm RVEA (theoretically, one can select any algorithm here), while the single-objective EA in the speculative interleaving and HK-RVEA algorithms is a standard generational genetic algorithm (GA) (its parameters are specified below). The study considers also RVEA optimizing in an environment without latencies as a kind of a baseline algorithm. That is, RVEA uses a maximum number of function evaluations for both objective functions, and thus should perform well in general. Having this baseline algorithm enables us to investigate the performance gap between optimizing with and without latencies as well as compare the performances of different algorithms relative to a 'control' algorithm and each other.

In addition to the effect of the latency, we study how correlations between the objectives affect the performance of the algorithms. The parameter settings in all the algorithms are kept the same:

1. latency: 1, 2, 4, 8, 10, 15
2. the expensive objective function: f_2
3. maximum number of function evaluations for f_1: 1000
4. maximum number of function evaluations for f_2: 1000/latency and 1000 when using RVEA
5. initial population size (also the initial size of the training data set in HK-RVEA): 50
6. single-objective EA: GA with a population size 50 and tournament selection of size 2
7. crossover: simulated binary crossover with 0.8 crossover probability and 20 distribution index
8. mutation: polynomial mutation with 1/(dimensions in the decision space) probability and 20 distribution index.
9. multiobjective EA in HK-RVEA: RVEA (same parameter settings as in [6])

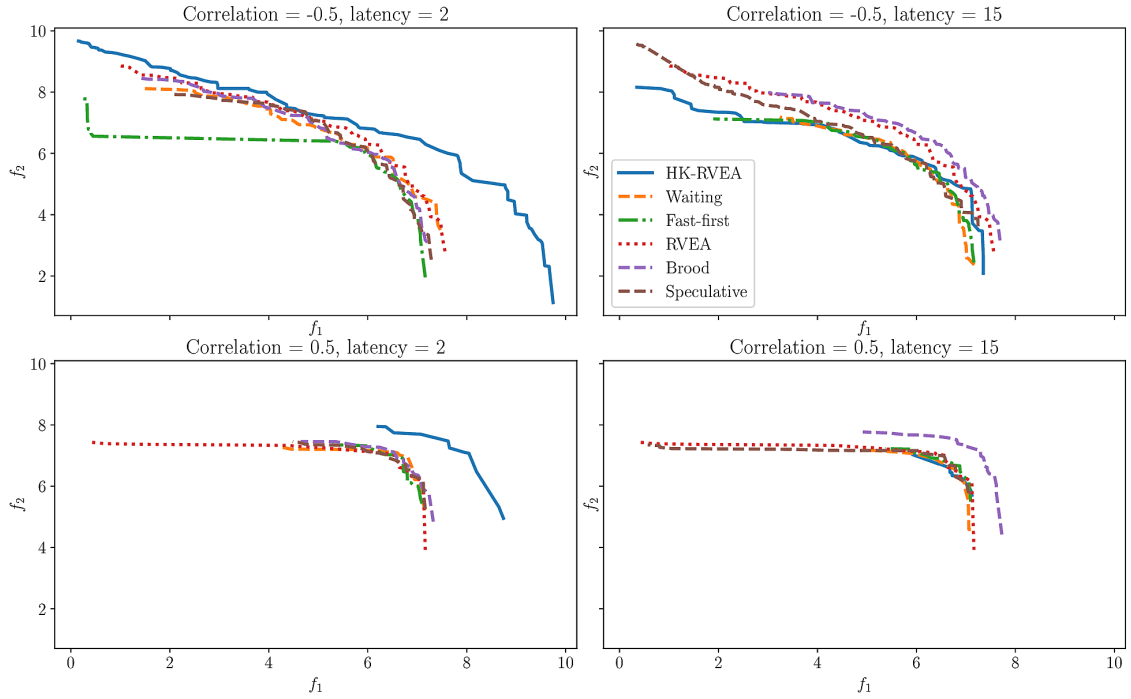

Figure 1: Median attainment surfaces of the cm-OneMax problem obtained by different algorithms. Note that both of the objectives are to be maximized.

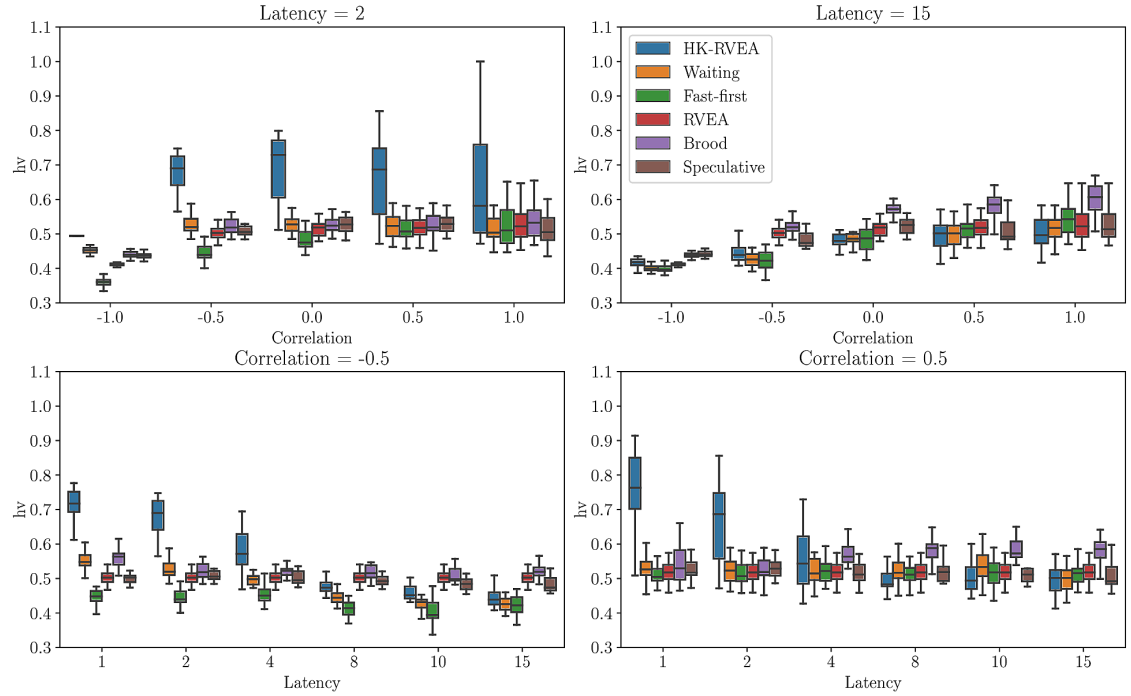

Figure 2: Median hypervolumes of the cm-OneMax problem obtained by different algorithms.

10. number of samples to update the surrogates: 3

11. number of independent runs: 21

We used 7 test problems from the DTLZ [11] and 7 problems from the UF [23] families as well as a continuous version of the mapped OneMax problem, which we call cm-OneMax, considered in [2] for binary search spaces. The cm-OneMax problem allows us to control the correlation among the two objective functions and thus enables us to investigate the sensitivity of different algorithms towards this problem feature. Assuming $n_1(x)$ to be the sum of all variable values in a candidate solution vector x, $n_2(y)$ the sum of all variable values in y, where y is a mapped version of solution vector x, then we can define the mapped continuous biobjective OneMax version (or cm-OneMax) as

$$f = (f_1, f_2) = (n_1(x), n_2(y)),$$

where $y_i = |x_i - map_i|, i = 1, \ldots, n$, n being the number of decision variables. The mapped value of a decision variable is $map_i \in [0, 1]$ and it is set independently for each decision variable $i = 1, \ldots, n$ (in our study we have $n = 10$) by flipping a coin biased by the degree of correlation $corr \in [-1, 1]$ one desires. That is, no correlation between the objectives corresponds to $corr = 0$, while complete positive and negative correlation corresponds to $corr = 1$ and $corr = -1$, respectively. For an intermediate correlation of $corr$, we set $map_i, i = 1, \ldots, n$ to zero with a probability of $(1.0 + corr)/2$.

Altogether, the experimental study considers 114 different test problems varying in latencies (applies to all test problems) and/or correlation between the objectives (cm-OneMax only). In the following, we use two key performance metrics, the hypervolume indicator [24] and attainment surfaces [12], to present and discuss the most representable and interesting results. We used the maximum of objective function values from all 21 runs with different latencies and correlations as the reference point in calculating the hypervolume.

4.2 Experimental results

In Figure 1, the median attainment surfaces of the cm-OneMax problem are shown for high and low correlations and latency values. On the top row, the correlation is -0.5 between the objectives with latencies 2 and 15. On the bottom row, the correlation is 0.5 and latencies 2 and 15. Note that both the objectives are to be maximized in this problem. As can be seen in the figure, HK-RVEA outperformed the other algorithms on a low latency for both low and high correlations. However, for a high latency, it did not perform well. This is due to the reason that at a high latency, the size of the training data set is smaller compared to the one at a low latency. Also, at a high latency, the number of function evaluations when doing single-objective optimization for the cheap objective function i.e. f_1 is significantly higher. Such a high number of function evaluations for one objective function does not provide a uniform distribution of samples for training and, therefore, poses a challenge for the performance of the surrogates for multiple objectives. Improving the performance of the surrogate-based algorithm on high latencies is considered as a future work.

On the other hand, the brood interleaving algorithm performs best for high latencies. This is because it evaluates a more diverse set of solutions on the cheap objective than HK-RVEA and speculative interleaving, while the expensive objective is evaluated. (In place where brood interleaving uses uniform selection and thus less selective pressure to generate solutions for evaluation on the cheap objective, both HK-RVEA and speculative interleaving employ a single-objective EA to drive the evolution.) The superior performance of brood interleaving over speculative interleaving for high latencies has also been observed in [2].

The waiting and fast-first algorithms behave in a similar way to the observations made in [2]: waiting does not show any bias towards the expensive or cheap objective but has a slower convergence, while fast-first is doing well on the cheap objective and poorly on the expensive objective (since this objective is not actually optimized by this algorithm).

When comparing all algorithms with the baseline algorithm RVEA, which optimizes the same problem but without latencies, it is clear that the presence of delays has a more severe impact on the performance if the objectives are negatively correlated. This is as expected because a more diverse population is needed in this case, which in turn can only be achieved by allowing more function evaluations.

We also plot the median hypervolume of the cm-OneMax problem obtained by the six different algorithms with different correlations and latencies in Figure 2. As can be seen on the top row,

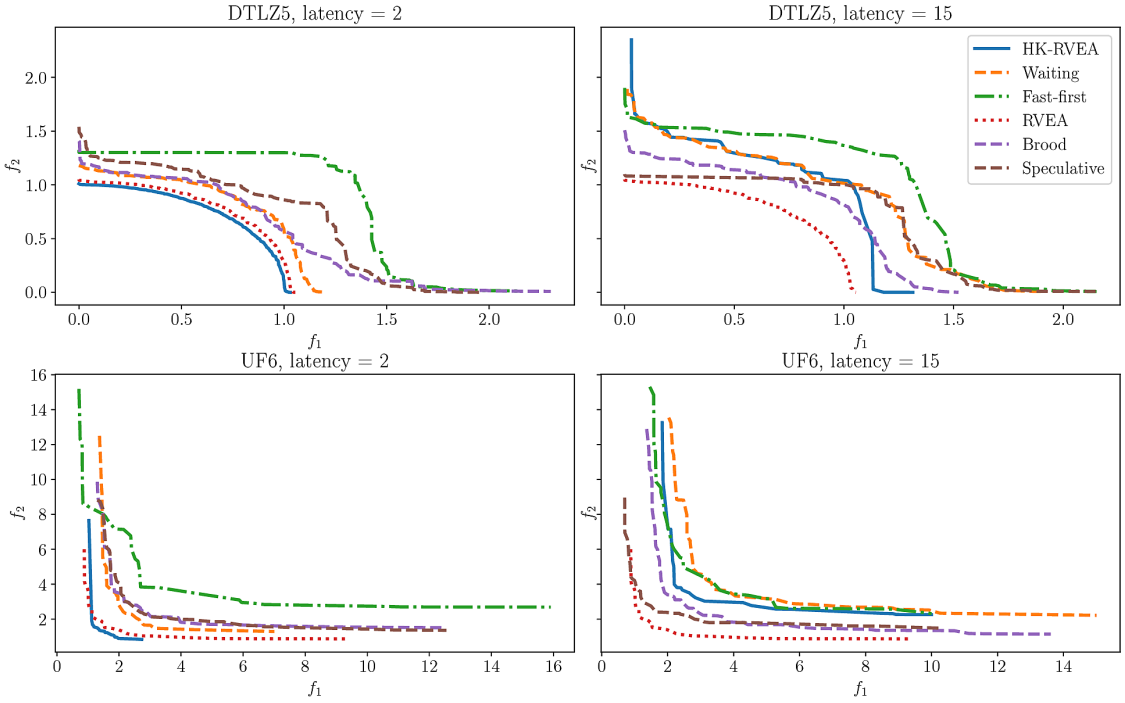

Figure 3: Median attainment surfaces of the DTLZ5 and UF6 problems obtained by different algorithms. RVEA is included for comparative purposes and it does not involve latencies.

the hypervolumes of HK-RVEA are bigger than the ones obtained by other algorithms at a low latency for all correlations and lower at a high latency. This is due to the size and the distribution of the training data set as mentioned above. One more key observation from the figure is that the variance is high for high correlations at a low latency when using the HK-RVEA algorithm. We will analyze the high variance in more details in the future.

On the bottom row, we show the box plots of the hypervolumes with different latencies at the correlations -0.5 (left) and 0.5 (right). The performance of HK-RVEA is good with low latencies but decreases with the increase in latencies. As indicated above in the attainment surface plots, the performance of brood interleaving improves as the latencies increase. Brood interleaving performs slightly better than speculative interleaving if correlations between objectives are negative because a more diverse population is more beneficial in this case. The algorithms waiting and fast-first rely heavily on the latency and correlation between the objectives. Waiting performs relatively better (i) the shorter the latency because the algorithm goes at the rate of the expensive objective, and (ii) the more the objectives are negatively correlated because here it seems to be more beneficial to maintain a diverse population without bias. Fast-first performs better the more the objectives are correlated positively because, in this setting, optimizing the cheap objective means to optimize the expensive objective too. The performance of fast-first does not depend on the latency because the algorithm goes at the rate of the cheap objective.

We also test with the DTLZ and UF problems, where the correlations among the objective functions cannot be controlled. Due to the limited space, we present only a subset of the results and more results including a table of hypervolume values and figures are available at http://www.mit.jyu.fi/optgroup/extramaterial.html. In Figure 3, we show median empirical attainment surfaces for the DTLZ5 and UF6 problems with latencies 2 and 15. As can be seen, the attainment surfaces of HK-RVEA are very close to the Pareto front at a low latency and similar results were obtained for the other DTLZ and UF problems. Moreover, the RVEA algorithm performs similar to HK-RVEA. This is due to the reason that the number of function evaluations when using RVEA is significantly higher for the expensive objective function (i.e. 1000 for both objective functions).

It is important to observe the efficiency and reliability of the algorithm with respect to different latencies. Therefore, we plot median hypervolumes of DTLZ5 and UF6 with different latencies in Figure 2. As can be seen, the performance of HK-RVEA is good with low latencies and starts

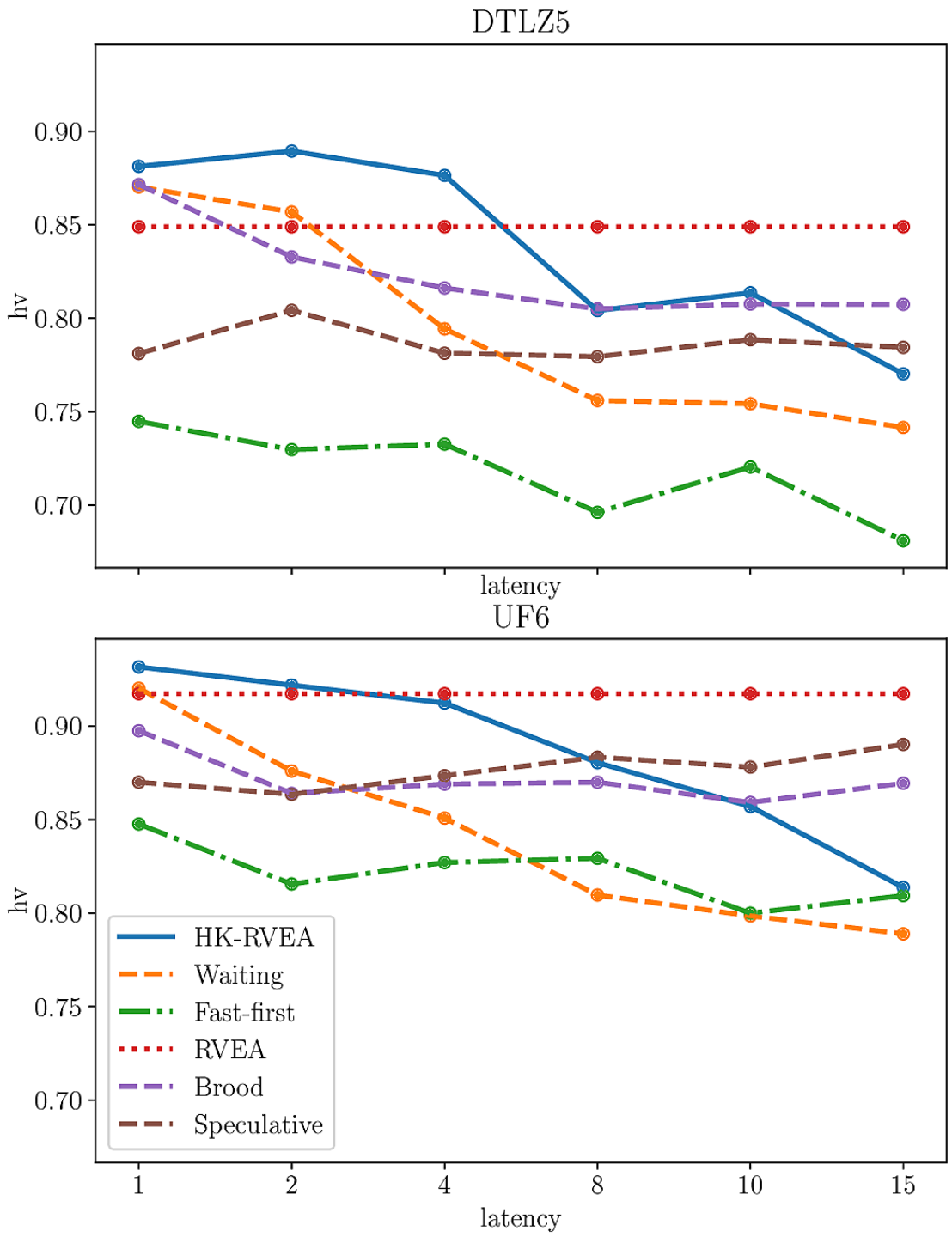

Figure 4: Median hypervolumes of DTLZ5 and UF6 problems on different latencies obtained by different algorithms.

deteriorating after a latency of four. Therefore, we suggest to use this algorithm for up to a latency of four (for such problems with two objectives and 10 variables). At the same time, it is difficult to generalize the efficiency of the algorithm because of different problem characteristics and dimensions. It may happen that at low dimensions, the algorithm will perform better, which will be analyzed in more details in the future. Overall, HK-RVEA shows promise whenever the latencies are not too big.

5 Conclusions

We have proposed a surrogate-assisted multiobjective evolutionary algorithm, termed HK-RVEA to handle computationally expensive multiobjective optimization problems that have different (i.e. heterogeneous) latencies (or evaluation times) among the objective functions. HK-RVEA uses Kriging models as surrogates and adapts the original mechanisms of K-RVEA for updating the surrogates to deal with the challenges arising from the different latencies in the objectives. In particular, the training and updating mechanism is driven by a single-objective EA, which interacts with the evolutionary multiobjective optimization algorithm used.

The efficiency of HK-RVEA was demonstrated on several biobjective benchmark problems with different latencies and correlations among the objective functions. A comparison with previously proposed methods that do not involve surrogates showed that HK-RVEA works well in cases with low latencies.

The good performance of HK-RVEA in coping with biobjective problems gives us hope that surrogate-assisted optimization can be generally viable for problems with different latencies. Some further attention must be devoted to topics like adapting the algorithm to cope with different latencies in high-dimensional objective and decision spaces, considering latencies in terms of both objectives and constraints, sensitivity of the algorithm with respect to the number of function evaluations, automated (online) tuning of algorithm parameters (e.g., the size of the training data set), comparing with other surrogate-assisted methods and developing suitable benchmark problems (e.g., tunable in terms of correlations in the objectives). Finally, we plan to employ and validate the developed algorithms on real problems with different latencies.

Acknowledgements
This work was partly supported by Tekes, the Finnish funding agency for innovation under the FiDiPro project DeCoMo (Chugh) and the Academy of Finland, grant 287496 (Ojalehto).

References

[1] R. Allmendinger, M. Emmerich, J. Hakanen, Y. Jin, and E. Rigoni. Surrogate-assisted multicriteria optimization: Complexities, prospective solutions, and business case. *Journal of Multi-Criteria Decision Analysis*, 24(1):5–24, 2017.

[2] R. Allmendinger, J. Handl, and J. Knowles. Multiobjective optimization: When objectives exhibit non-uniform latencies. *European Journal of Operational Research*, 243(2):497–513, 2015.

[3] R. Allmendinger and J. Knowles. 'hang on a minute': Investigations on the effects of delayed objective functions in multiobjective optimization. In R. P. et al., editor, *Proceedings of the 7th International Conference on Evolutionary Multi-Criterion Optimization*, pages 6–20, Berlin, Heidelberg, 2013. Springer.

[4] S. Bagheri, W. Konen, R. Allmendinger, J. Branke, K. Deb, J. Fieldsend, D. Quagliarella, and K. Sindhya. Constraint handling in efficient global optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 673–680. ACM, 2017.

[5] D. Brockhoff and E. Zitzler. Objective reduction in evolutionary multiobjective optimization: Theory and applications. *Evolutionary Computation*, 17(2):135–166, 2009.

[6] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 20:773–791, 2016.

[7] T. Chugh, Y. Jin, K. Miettinen, J. Hakanen, and K. Sindhya. A surrogate-assisted reference vector guided evolutionary algorithm for computationally expensive many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 22:129–142, 2018.

[8] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen. A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms. *Soft Computing*, 2018.

[9] T. Chugh, K. Sindhya, K. Miettinen, J. Hakanen, and Y. Jin. On constraint handling in surrogate-assisted evolutionary many-objective optimization. In J. e. a. Handl, editor, *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature*, pages 214–224. Springer, 2016.

[10] T. Chugh, K. Sindhya, K. Miettinen, Y. Jin, T. Kratky, and P. Makkonen. Surrogate-assisted evolutionary multiobjective shape optimization of an air intake ventilation system. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1541–1548. IEEE, 2017.

[11] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 825–830. IEEE, 2002.

[12] C. Fonseca and P. Fleming. On the performance assessment and comparison of stochastic multiobjective optimizers. In *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, pages 584–593. Springer, 1996.

[13] J. Hakanen, K. Sahlstedt, and K. Miettinen. Wastewater treatment plant design and operation under multiple conflicting objective functions. *Environmental Modelling & Software*, 46:240–249, 2013.

[14] G. Hornby. 'alps': the age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pages 815–822. ACM, 2006.

[15] J. Knowles. ParEGO: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10:50–66, 2006.

[16] A. Lewis, S. Mostaghim, and I. Scriven. Asynchronous multi-objective optimisation in unreliable distributed environments. In A. Lewis, S. Mostaghim, and M. Randall, editors, *Biologically-Inspired Optimisation Methods*, pages 51–78. Springer, 2009.

[17] D. Lim, Y. Jin, Y.-S. Ong, and B. Sendhoff. Generalizing surrogate-assisted evolutionary computation. *IEEE Transactions on Evolutionary Computation*, 14(3):329–355, 2010.

[18] S. Lophaven, H. Nielsen, and J. Sondergaard. DACE: a MATLAB Kriging toolbox. Technical report, Technical University of Denmark, 2002.

[19] A. Mogilicharla, T. Chugh, S. Majumdar, and K. Mitra. Multi-objective optimization of bulk vinyl acetate polymerization with branching. *Materials and Manufacturing Processes*, 29:210–217, 2014.

[20] B. G. Small, B. McColl, R. Allmendinger, J. Pahle, G. López-Castejón, N. Rothwell, J. Knowles, P. Mendes, D. Brough, and D. Kell. Efficient discovery of anti-inflammatory small-molecule combinations using evolutionary computing. *Nature Chemical Biology*, 7:902–908, 2011.

[21] S. Van Buuren. *Flexible imputation of missing data*. CRC press, 2012.

[22] T. Walters. Repair and brood selection in the traveling salesman problem. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature*, pages 813–822. Springer, 1998.

[23] Q. Zhang, A. Zhau, S. Zhao, P. Suganthan, W. Liu, and S. Tiwari. Multiobjective optimization test instances for the CEC 2009 special session and competition. Technical Report CES-487, University of Essex/ Nanyang Technological University, Essex, U.K./Singapore, 2009.

[24] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2):117–132, 2003.